World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:1, No:5, 2007

Robot Path Planning in 3D Space Using Binary Integer Programming

Ellips Masehian, and Golnaz Habibi

***Abstract*—**This paper presents a novel algorithm for path planning of mobile robots in known 3D environments using Binary Integer Programming (BIP). In this approach the problem of path planning is formulated as a BIP with variables taken from 3D Delaunay Triangulation of the Free Configuration Space and solved to obtain an optimal channel made of connected tetrahedrons. The 3D channel is then partitioned into convex fragments which are used to build safe and short paths within from Start to Goal. The algorithm is simple, complete, does not suffer from local minima, and is applicable to different workspaces with convex and concave polyhedral obstacles. The noticeable feature of this algorithm is that it is simply extendable to *n*-D Configuration spaces.

***Keywords*—**3D C-space, Binary Integer Programming (BIP), Delaunay Tessellation, Robot Motion Planning.

I. Introduction

THE Robot motion planning problem first emerged in 1960's when researchers were trying to endow the primitive robotic mechanisms with autonomy and intelligence to find collision-free paths from start to goal. The motion planning problem however proved to be PSPACE-hard and NP-complete [4]. The dimension of the environment is a decisive factor influencing the complexity of motion planning. In Fact, what makes Motion Planning hard is the dimensionality of the C-space, i.e., the space of all possible motions of the robot. Planning in 2D environments is the easiest one, while the 3D and higher-dimensional problems call for a challenge.

In this paper a new solution is proposed for the path planning problem for a robot in 3D Configuration Space. The presented model is a novel composition of known path planning methods and linear algebra techniques. First, we will briefly review some path planning techniques relevant to our work. Next, the model is presented in its 3 main phases: workspace tessellation, optimal channel finding, and path finding. The last sections of the paper deal with experiments and comparisons, as well as further discussions.

A. Motion Planning

Many solution methods for robot motion planning are variations of a few general approaches: *Roadmap*, *Cell Decomposition*, and *Potential Fields* methods, which are broadly surveyed in [6],[7] and [12]. Also, heuristic methods have gained a wide popularity in recent years [9].

Authors are with Faculty of Engineering, Tarbiat Modares University, Tehran, Iran.

The *Visibility Graph* is a roadmap which is the collection of lines in the free space connecting the feature of an object, usually vertices, to that of another object. For higher-dimensional spaces than 2D, the visibility graph can be constructed from the vertices of polyhedra, but the shortest path then no longer lies in the visibility graph. Therefore, visibility graphs are impractical in for 2D motion planning.

Voronoi Diagram is defined as the set of points equidistant from two or more object features, and is used widely in motion planning, especially in low dimensions, such as [3] and [14]. Generalizing Voronoi Diagrams to higher spaces is a challenging problem and there are some works in this respect as [16], [6], and [5].

The Voronoi diagram has a dual problem, the *Delaunay Triangulation* DT(S), which is obtained with a line segment between any two points p and q in the set of points S, for which a circle C exists that passes through p and q and does not contain any other site of S in its interior or boundary [1]. The edges of DT(S) are called *Delaunay edges*.

The Delaunay Triangulation can also be extended to 3 and higher dimensional spaces, and so there are Delaunay Edges and Faces. In this case, the workspace is decomposed to tetrahedrons such that the sphere passing through the vertices of each tetrahedron doesn't contain other vertices (Fig. 1). It is also the dual of 3D Voronoi Diagram [2].

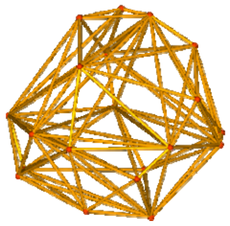


Fig. 1 Tetrahedrons are formed by 3D Delaunay Triangulation

In *Potential Fields* (PF) method, the robot is treated as a point represented in C-space as a particle under the influence of an artificial potential field which can be defined over free space as the sum of an attractive potential pulling the robot toward the goal configuration, and repulsive potentials pushing the robot away from the obstacles [12]. Although PF can easily be extended to 3D spaces, the problem of falling in local minima exists. In [17] a PF model for path planning is presented where the steady state heat transfer (as potential function) is simulated with variable thermal conductivity.

The *Cell Decomposition* approach divides the free configuration space (C_{free}) into a set of non-overlapping cells. The adjacency relationships between cells are represented by a

Connectivity Graph. The graph is then searched for a collision free path. A solution is obtained by finding the cells that contain the initial and final configurations and then connecting them using a sequence of connected cells. If the union of these cells is exactly equal to C_{free}, then the approach is called *exact* cell decomposition (which uses the object boundaries to generate the cell boundaries, hence object dependent), otherwise, it is called *approximate* (which partitions the C-space into cells of a simple shape and then computes if the cell is in C_{free}, hence object independent) [10]. The idea of using Cell decomposition for RMP was used in [11]. In our proposed method, the C_{free} is exactly decomposed into Delaunay Tetrahedrons.

B. *Linear Programming*

Linear Programming (LP) is an important field of optimization for several reasons. Many practical problems in operations research can be expressed as linear programming problems. LP is heavily used in very different disciplines, such as microeconomics, business management, portfolio and finance management, inventory management, production planning, resource allocation, food blending, etc., typically to maximize the output, or minimize the costs of the system. The standard form of an LP is:

$$\text{Maximize } C^T x \text{ , Subject to } A \cdot x \leq b,\ x \geq 0 \qquad (1)$$

Binary Integer Programming (BIP) is a special case of LP where the variables have only binary 0-1 integer values. BIP problem is NP-hard and can be solved by methods like Branch and Bound (BNB) and Gomory Cut algorithms [8].

Compared to other applications, linear programming has been implemented very few in motion planning. However, a number of works exist, such as [5] in which methods to solve collision-free fuel-optimal path- and motion-planning problems for the reconfiguration of spacecraft formations are presented. The motion planning problem is formulated as a parameter optimization problem, the trajectory being parameterized by the spacecraft positions and velocities at a set of waypoints. Big-M relaxation is then used to formulate the parameter optimization as a mixed-integer linear program (MILP). Another work is [15] where mechanisms used to design a path planner with real-time and long-range capabilities are presented. The approach relies on converting the optimal control problem into a parameter optimization one whose horizon can be reduced by using a global cost-to-go function to provide an approximate cost for the tail of the trajectory. Thus only the short-term trajectory is being constantly optimized online based on a MILP formulation.

II. Overview of the New Model

The presented path planner is a generalization of the work in [13] where the workspace is extended from 2D to 3D and *n*-D. It lies within the category of Cell Decomposition approach. However, unlike the usual procedure in Decomposition-based models, which builds a connectivity graph and then searches it to find an optimal channel of cells, this model implements a mathematical programming for finding the optimal channel. The general phases of the algorithm are the following:

1) The C_{free} is tessellated through the 3D Delaunay Triangulation such that it is exactly decomposed into a set of tetrahedrons (Section III).

2) Each tetrahedron is associated with a variable that is incorporated in a 0-1 Integer Programming model with an appropriate objective function (Section V). The solution gives a set of connected tetrahedrons forming a channel.

3) The channel is further segmented into a number of convex fragments, and a safe path is calculated passing through the medians of the convex fragments (Section VI).

The following sections describe these phases in detail.

III. Workspace Tessellation

The first step for path planning is to express the workspace (or C-space) as a mathematical formulation. For this purpose, the 3D workspace is decomposed into arrangements of cells through 3D Delaunay Triangulation. The reason for choosing Delaunay triangulation for tessellating the workspace is its many useful optimization properties discussed in [1].

The input information for the 3D triangulation consists of the coordinates of all obstacles' vertices and the inner corners of the workspace's cuboid border. The Delaunay algorithm then builds a connected network of these points, forming a set of tetrahedrons. These tetrahedrons lie both inside the obstacles and the C_{free}. By applying a simple checking algorithm, all the tetrahedrons inside obstacles can be identified and omitted from the tetrahedrons set, leaving 'free' tetrahedrons that build the C_{free}.

Fig. 2(a) shows the result of the above operations. In order to avoid long and thin tetrahedrons or ones with wide extension (especially near borders), the borders can be segmented into equal intervals. The result of tessellating this variation is shown in Fig. 2(b). Note the increased regularity and 'fatness' of tetrahedrons, despite their larger number.

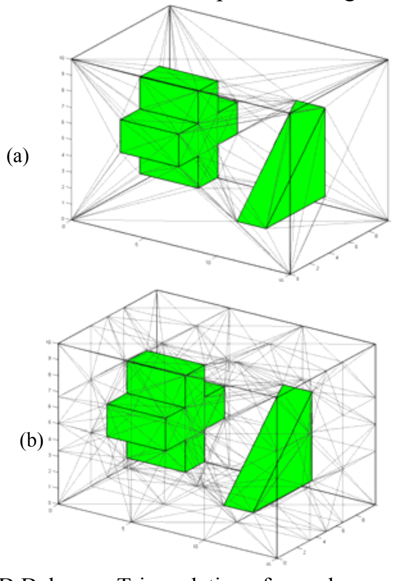


Fig. 2 3D Delaunay Triangulation of a workspace with (a) unsegmented, and (b) segmented borders and obstacles

IV. Optimality Criteria

The presented algorithm determines an optimal channel among all possibilities such that the objective function of the optimization problem is minimized regarding to constraints of the problem. So, it is very crucial to establish sound and robust criteria for defining the objective function.

The objective function is the product of two matrices: a *Weighting* matrix and the *Variables* matrix (discussed in Section V). We worked out five different optimality criteria as follows:

1) Number of Tetrahedrons: This kind of weighting merely depends on the number of tetrahedrons, regardless of their size. The weighting matrix (W_N) is a row vector of 1s, and so the system selects a Start-to-Goal channel with minimal number of constituting tetrahedrons. It is useful when the tetrahedrons are almost equal in size.

2) Volume of Tetrahedrons: In this criterion, larger tetrahedrons have larger weights, and the objective function turns into the weighted sum of variables. The algorithm searches for a channel with the smallest volume. This criterion (W_V) is efficient when the tetrahedrons have the same size or at least are similar.

3) Surface of Tetrahedrons: According to this criterion, larger tetrahedrons have larger weights. The algorithm searches for a channel with the smallest total surface. This criterion (W_S) is efficient when the tetrahedrons are symmetric or similar.

4) The Diameter of Insphere of Tetrahedrons: An idea for defining a median path inside the channel is the diameter of insphere (inscribed sphere) of each tetrahedron. This weighting function (W_I) is more efficient when tetrahedrons with more than two free faces are symmetric. Also, in order to care about the distances of Start and Goal points to their neighboring tetrahedrons, these distances are added to the diameter of inscribed sphere of neighboring triangles as their weights.

5) The Median Length of Channel: The most reliable and effective weighting function we found is the median length of channel (W_M) which is defined by the median length of each tetrahedron. But this length is not unique since it depends on the size of tetrahedrons which are selected and the channel passes through. Therefore, it is reasonable to define the median length of each tetrahedron regarding to its selected neighbors. In this case the weighting function is as follows:

$$F = \sum_{i=1}^{n} \sum_{j<k \in N(t_i)}^{N_i} w_{ijk} t_i t_j t_k \tag{2}$$

where $N(t_i)$ is the set of indices of neighbors of t_i, and w_{ijk} is the median length of path through tetrahedron t_i. For instance, to calculate the median length of tetrahedron t_1 with its neighbors being t_2, t_3, and t_4, we have:

$$W_1 = w_{123} \times t_1 t_2 t_3 + w_{124} \times t_1 t_2 t_4 + w_{134} \times t_1 t_3 t_4.$$

In other words, if t_1 is selected while t_2 and t_3 are also selected from among t_1's neighbors, then the median length weight for t_1 would be $W_1 = W_{123}$.

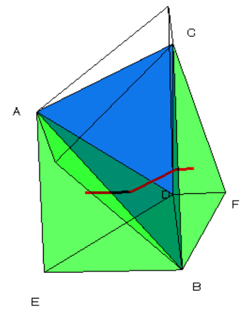


Fig. 3 A channel with its median line, when tetrahedron ABCD is selected with its neighbors, ABDE and CDBF

Although this weighting function seems more efficient, this weighting function is nonlinear and it is in cubic form. So it is not convex and it doesn't guarantee the optimal solutions. Moreover, it takes a long time to solve the problem with this weighting function.

It should be noted that a weighted combination of these weighting methods can also be applied as

$$W = \alpha_1 \times W_N + \alpha_2 \times W_V + \alpha_3 \times W_S + \alpha_4 \times W_I + \alpha_5 \times W_M. \tag{3}$$

V. BIP Formulation and Solution

In this phase a minimization problem with linear constraints is developed. The objective function to be minimized is the weighted sum of the variables representing all tetrahedrons in C_{free}, which reflects a measure of optimality for the resulting channel. Following is a basic formulation of the problem.

The path planning problem can be modeled as a 0-1 Binary Integer Programming (BIP), with variables defined as

$$t_i = \begin{cases} 1 & \text{if tetrahedron } i \text{ is selected} \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

In order to guarantee a continuous channel from the Start to Goal points, tetrahedrons building the trajectory channel must satisfy the following requirements:

1. The Start and Goal tetrahedrons (t_S and t_G, respectively) must be selected.
2. If any tetrahedron (other than t_S and t_G) is selected, two of its adjacent tetrahedrons must also be selected (continuity condition).
3. Each of the Start and Goal tetrahedrons must have only one selected adjacent tetrahedrons (loop avoidance condition).
4. To avoid looping, only two adjacent tetrahedrons of tetrahedrons with three free edges must be selected.

With the above considerations, the BIP model for finding the optimal channel will be:

$$\text{Minimize } J = W^T.T$$

Subject to:

$$\begin{cases} t_1 + \sum_{j=1}^{n_1} t_1^j \geq 3t_1 \\ \quad\cdots \\ t_{k-2} + \sum_{j=1}^{n_{k-2}} t_{k-2}^j \geq 3t_{k-2} \end{cases} \tag{5}$$

$$\begin{cases} t_S + \sum_{j=1}^{n_S} t_S^j = 2t_S \\ t_G + \sum_{j=1}^{n_G} t_G^j = 2t_G \end{cases} \tag{6}$$

$$\begin{cases} t'_1 + \sum_{j=1}^{3} t'^j_1 \leq 3 \\ \quad\cdots \\ t'_p + \sum_{j=1}^{3} t'^j_p \leq 3 \end{cases} \tag{7}$$

$$\begin{cases} t''_1 + \sum_{j=1}^{4} t''^j_1 \leq 3 \\ \quad\cdots \\ t''_q + \sum_{j=1}^{4} t''^j_q \leq 3 \end{cases} \tag{8}$$

$$t_S = 1,\ t_G = 1,\ t_i \in \{0, 1\},\ i \notin \{S, G\}.$$

In this model, W is the column vector of weights (calculated from (3) with desired coefficients α_i), and T is the vector of variables $[t_i, \ldots, t_k]^T$. The t_i^j is the jth adjacent tetrahedron of t_i, and $j = 1, \ldots, n_i$, in which $n_i = 1, 2$, or 3.

In (7), $\{t'_t\}$ $(t = 1, \ldots, p)$ is the set of p tetrahedrons with three free adjacent tetrahedrons and t'^j_t is the jth adjacent tetrahedron of tetrahedron t'_t.

In (8), $\{t''_f\}$ $(f = 1, \ldots, q)$ is the set of q tetrahedrons with four free adjacent tetrahedrons and t''^j_f is the jth adjacent tetrahedron of tetrahedron t''_f.

There is one constraint for each tetrahedron, and so the path planning problem turns into an optimal planning with $k+p+q$ constraints ($k-2+p+q$ inequalities and 2 equalities), with an optimality criterion defined by objective function J.

VI. Path Calculation

Upon finding the optimal channel, the next phase is to find an appropriate trajectory through it. In 2D dimension the shortest path can be found via the Visibility Graph method, but in higher dimensions this method isn't as well as the 2D, so it is rarely used in 3D or higher dimensions.

Another simple path is through the centers of gravity of each tetrahedron in the channel. However, it usually takes a zigzag form, which is not suitable for robot motion and acts poorly in the presence of compact tetrahedrons (dashed line in Fig. 4)

The typical method for path finding in the classic Cell Decomposition approach is to connect the middles of free faces of cells in the optimal channel. There are two options for connecting the middles of common faces of neighboring tetrahedrons building the channel: The first approach is to connect the gravity centers of the common faces (the dotted line in Fig. 4). Another way is to connect the centers of inscribed circle of faces. These obtained paths are mostly the same and they have equal lengths. While this method yields much smoother and shorter path than the path through the centers of gravity, it usually travels unduly lengthy distances and has a zigzag form such that it is considerably longer than the shortest path.

A. *Path through the Middles of Convex Fragments*

A more efficient and smoother trajectory can be suggested is the path going through the middle of convex fragments of the optimal channel.

At first, beginning from the Start tetrahedron, neighboring tetrahedron in the channel are appended until reaching a tetrahedron that makes the appended set concave. By this, the largest possible convex fragment is formed. The next convex fragment starts from the next tetrahedron. Then, a path is calculated based on the midpoints of face borders of convex fragments.

The obtained trajectory is much shorter than the other paths discussed above. Besides, the fact that a convex polyhedron contains any line connecting any two points in the polyhedron, guarantees that the path is definitely located in the channel and does not cross the obstacles. Fig. 4 shows path planning based on three types of trajectories: 1) centers of gravity, 2) middles of common faces, and 3) middles of convex fragments. It is shown that the trajectory based on middle edges of the convex fragments is the most reliable and shortest path among them.

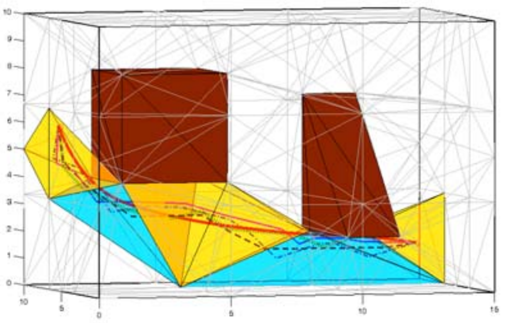


Fig. 4 Path planning with the optimal channel and five trajectories based on 1) centers of gravity (dashed), 2) centers of inspheres (dash-dotted), 3) centers of common faces (dotted), 4) middles of common faces of 5 convex fragments (thin solid), and 6) B-spline curve for middles of 5 convex fragments common faces (thick solid)

B. *B- Spline Curves*

For smoothing the paths mentioned in 3D B-splines can be used with control points being can be one of points used for other paths. It is obvious that the B-spline based on path through convex fragments is more smooth and shortest curve.

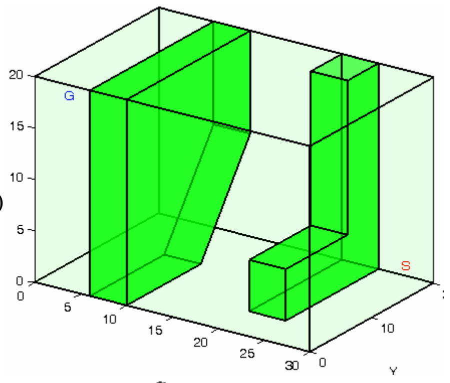
For designing a shorter and smoother trajectory, sometimes the *concaveness* of a fragment can be ignored. The amount of concaveness can be measured by the *degree of concaveness*, which is related to the degree of the concave angle in the fragment. This angle is formed between faces of the newly added tetrahedron and the existing fragment. Thus, the greater the degree of the concave angle, the more the degree of concaveness. The range of the degree of concaveness is $180 \leq \alpha < 360$, in which $\alpha = 180$ represents the minimum (i.e. no) concaveness. For $\alpha \leq 180$, the polyhedron is convex, and the degree $\alpha = 360$ yields the maximum concaveness.

When the α takes a value between 180 and 360 degrees, some amount of concaveness is tolerated, and the fragments are allowed to be somewhat concave. In return, this causes a shorter path passing through the middles of fragments. Note that with large tolerances, the path may be out of the channel and cross the obstacles. So the shortness and safety of the path must be balanced.

The cumulative angle error, as a result of ignoring the concaveness for shorter paths should be considered too. The threshold of the cumulative error is up to the designer, and smaller thresholds lead to more conservativeness.

VII. Experiments and Comparison

In this section, the whole procedure of 3D path planning with the new method is illustrated. At first, all 3D obstacles, as well as the Start and Goal points are defined by the user. Fig. 5(a) shows a sample workspace with 2 obstacles (a convex and a concave) having 30 vertices. The workspace is then triangulated by 3D Delaunay Triangulation, which took 0.005 seconds with a 2.3 GHz PC, shown in Fig. 5(b).

(a)


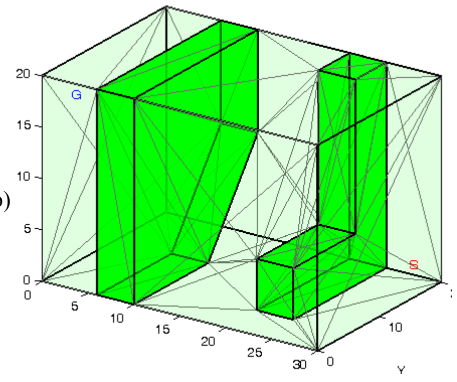
(b)


Fig. 5a) Workspace, and b) 3D Delaunay Triangulation of the workspace

The next stage is to build the BIP model and solve it. Fig. 6 illustrates the obtained optimal 3D channel. It took 0.038 seconds to find the channel. In this example, the optimality criteria were the number of tetrahedrons and the median length of the channel. Also, a spline function was implemented to smooth the path. Fig. 7 shows the spline path within the optimal channel by using BIP.

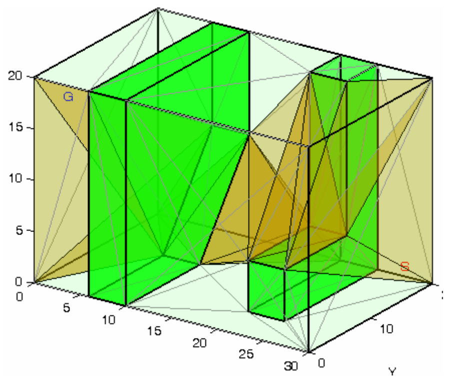


Fig. 6 The optimal channel by binary integer programming

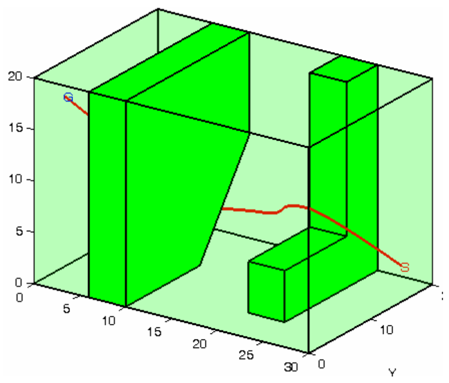


Fig. 7 B-Spline Path within the optimal channel

We implemented the proposed algorithm in some workspaces, and compared with other methods. A typical problem took 0.5 seconds to be solved by BIP (Fig. 8) and 1.23 seconds by Voronoi graph (Fig. 9).

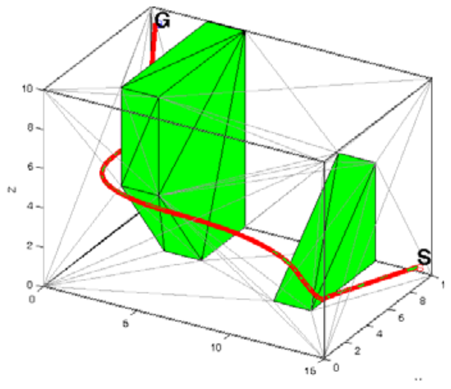


Fig. 8 Path planning using binary integer programming

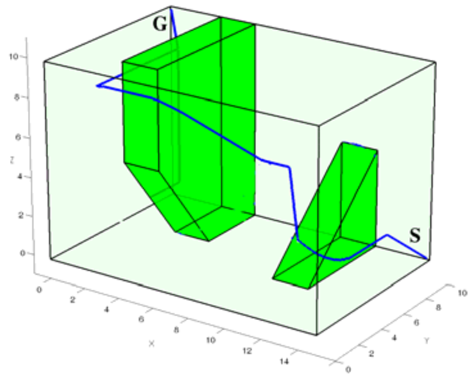


Fig. 9 Path planning using Voronoi Graph

VIII. Extension to Higher Dimensions

An interesting feature of the new algorithm is the ease of its extension to high dimensional workspaces. Since the algorithm's variables are taken from the cells resulting from Delaunay Triangulation, the cells can be triangles (in 2D), tetrahedrons (in 3D), or *n*-polytopes (in *n*-D). The constraints remain the same in higher dimensions; the only difference is in the maximum number of free neighbors for each *n*-polytope, which is *n*+1 for *n*-D space. The BIP formulation for cells other than Start and Goal cells will be

$$\begin{cases} t'_1 + \sum\limits_{j=1}^{3} t'^{\,j}_1 \le 3 \\ \quad\vdots \\ t'_p + \sum\limits_{j=1}^{3} t'^{\,j}_p \le 3 \end{cases} \tag{9}$$

$$\cdots$$

$$\begin{cases} t_1^{(n-1)} + \sum\limits_{j=1}^{n+1} t_1^{(n-1)j} \le 3 \\ \quad\vdots \\ t_{p_n}^{(n-1)} + \sum\limits_{j=1}^{n+1} t_{p_n}^{(n-1)j} \le 3 \end{cases} \tag{10}$$

In (10), $\{t_p^{(d)}\}$ ($p = 1, \ldots, p_n$, $d = 2, \ldots, n$) is the set of p tetrahedrons with d+1 free adjacent *n*-polytopes, and $t_p^{(d)j}$ is the jth adjacent *n*-polytope of the *n*-polytope $t_p^{(d)}$.

IX. Discussion and Conclusion

In order to analyze the time complexity of the algorithm, we consider its three phases separately. The time complexity of constructing a 3D Delaunay Triangulation is $O(n\log n)$, n being the number of all obstacle vertices [12]. Since we have binary integer variables, the complexity of BIP solution depends on the number of variables (i.e. tetrahedrons). The number of tetrahedrons has a linear order of n. On the other hand, the branching of the tetrahedrons' connectivity graph occurs only at those tetrahedrons that have more than 3 neighbors. These tetrahedrons are corresponding to Voronoi graph vertices and their number is in $O(m)$, m being the number of obstacles. By using the Depth-Best-First search (expand in depth and then the best solution), the complexity of BIP solution becomes $O(3^m)$, and so is independent of the number of vertices. The time complexity of path finding is linear in the number of tetrahedrons building the optimal channel.

A very important property of this method is its robustness to workspace complexities and local minima. Since tetrahedrons located at the bottom of a 'minimum well' do not satisfy the condition (7), they are never selected by the solver, and thus never appear in the final solution. These conditions also prevent the selection of all three neighbors of a tetrahedron, which forms a loop.

References

[1] F. Aurenhammer, and R. Klein, "Voronoi diagrams", in J. Sack and G. Urrutia (eds), *Handbook of computational geometry*, Elsevier Science Pub., 2000.

[2] Z. Bao, "Delaunay Triangulations and Voronoi Diagrams on Non-Planar Topologies," *CS20c Final Report*, California Institute of Technology, 1998, http://www.ugcs.caltech.edu/ ~jbao/Voronoi.pdf.

[3] J. F. Canny, "A Voronoi method for the piano movers' problem", in *Proc. ICRA* 1985, pp.530-535.

[4] J. F. Canny, *The complexity of robot motion planning*, The MIT press, Cambridge, Mass., 1988.

[5] B. Cetin, M. Bikdash, and F.Y. Hadaegh, "Hybrid mixed-logical linear programming algorithm for collision-free optimal path planning", *IET Control Theory & Applications,* 2007, Vol. 1, Issue 2, pp. 522-531.

[6] H. Choset, and J. Burdick, "Sensor-based exploration: the hierarchical generalized Voronoi graph", *Int. J. of Robotics Research*, Vol. 19, No. 2, (2000), pp. 96-125.

[7] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*, MIT Press, Boston, 2005.

[8] N. Christofides, A. Mingozzi, P. Toth, and C. Sandi, *Combinatorial Optimization*, John Wiley, 1979.

[9] C.Eldershaw," Heuristic Algorithms for robot motion planning,," Ph.D. Dissertation, the University of Oxford ,2001.

[10] Y. K. Hwang and N. Ahuja, "Gross motion planning - a survey", *ACM Comp. Surveys*, 24(3), (1992), pp. 219-291.

[11] J. M. Keil and J. R. Sack, "Minimum decompositions of polygonal objects" in G. T. Toussaint (ed.) *Computational Geometry*, North-Holland, Amsterdam, 1985, pp. 197-216.

[12] J. C. Latombe, *Robot motion planning*, Kluwer Academic publishers, London, 1991.

[13] G. Habibi, "Mobile Robot Path Planning Using Binary Integer Programming and Linear Matrix Inequalities (LMIs)," submitted to IEEE Int. Conf. Intell. Rob. Sys. (IROS) 2007

[14] N. S. V. Rao, N. Stolzfus, and S. S. Iyengar, "A 'retraction' method for learned navigation in unknown terrains for a circular robot", *IEEE Trans. Rob. Autom.,* Vol. 7, No. 5, (1991), pp. 699-707.

[15] O. Toupet, "Real-time path-planning using mixed integer linear programming and global cost-to-go maps", M.S. thesis, Dept. Aeronautics and Astronautics, MIT, February 2006.

[16] J. Vleugels and M. Overmars, "Approximating generalized Voronoi diagrams in any dimension," Technical report No. UU-CS-1995-14, Utrecht University, May 1995.

[17] Y. Wang and G. S. Chirikjian, "A new potential field method for robot path planning," in Proc. IEEE Int. Conf. Rob. Autom., 2000, pp. 977-982.